# Report about Neural Networks

Nikolaev Alexander, group 5130203/20102

## 1 Perceptron

### 1.1 Model Architecture

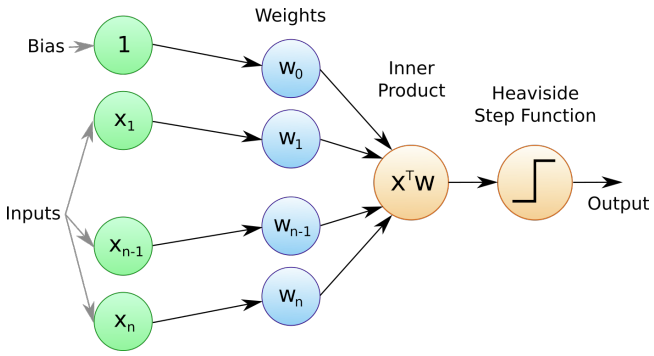


Figure 1: Perceptron Model Architecture

### 1.2 Vector Representation of Data

Let $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ be the input vector and $y \in \{0, 1\}$ be the output label.

### 1.3 Mathematical Formulation

The linear combination can be expressed as: $[z = \mathbf{w}^T \mathbf{x} + b]$ where $\mathbf{w}$ is the weight vector and $b$ is the bias.

The activation function used in a Perceptron is: $\hat{y} = \begin{cases} 1 & , z > 0 \\ 0 & \text{otherwise} \end{cases}$

### 1.4 Explanation of gradient descendent algorithm

Gradient descent is an optimization algorithm used to minimize the loss function by iteratively updating the weights and biases

### 1.5 Gradient Descent Algorithm

The Perceptron updates weights using the following rule: $\mathbf{w} \leftarrow \mathbf{w} + \eta(y - \hat{y})\mathbf{x}$ where $\eta$ is the learning rate.

The bias update is: $b \leftarrow b + \eta(y - \hat{y})$

## 2 Logistic Regression

### 2.1 Model Architecture

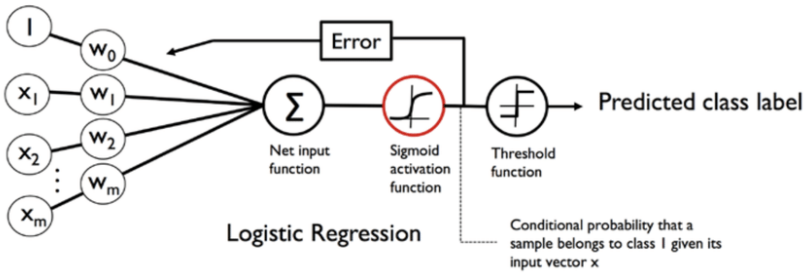


Figure 2: Logistic Regression Model Architecture

### 2.2 Vector Representation of Data

Let $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ be the input vector and $y \in \{0, 1\}$ be the output label.

### 2.3 Mathematical Formulation

The linear combination is given by: $z = \mathbf{w}^T\mathbf{x} + b$

The activation function (sigmoid function) is: $\hat{y} = \sigma(z) = \frac{1}{1+e^{-z}}$

The loss function (cross-entropy loss) is: $L(y, \hat{y}) = -ylog(\hat{y}) + (1-y)log(1-\hat{y})$

### 2.4 Explanation of gradient descendent algorithm

As with the perceptron, the gradient descent method is used to minimize the loss function by updating the weights and biases.

### 2.5 Gradient Descent Algorithm

The weights are updated as follows: $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla L$

The gradient of the loss with respect to weights is: $\nabla L = (\hat{y} - y)\mathbf{x}$

The bias update is: $b \leftarrow b - \eta(\hat{y} - y)$

## 3 Multilayer Perceptron

### 3.1 Model Architecture

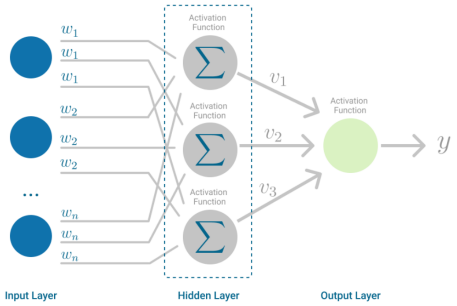


Figure 3: Multilayer Perceptron Model Architecture

### 3.2 Vector Representation of Data

Let $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ be the input vector and $y$ be the output label.

### 3.3 Mathematical Formulation

For each layer $l$, the linear combination is: $z^l = \mathbf{w}^l a^{l-1} + b^l$ where $a^{l-1}$ is the activation from the previous layer.

The activation function (e.g., ReLU or sigmoid) can be represented as: $a^l = f(z^l)$

The loss function can be similar to logistic regression: $L(y, \hat{y}) = -[ylog(\hat{y}) + (1-y)log(1-\hat{y})$

### 3.4 Explanation of gradient descendent algorithm

The MLP employs backpropagation to calculate the gradients of the loss function concerning weights and biases, facilitating efficient updates throughout the training process.

### 3.5 Gradient Descent Algorithm

Weights are updated as follows: $\mathbf{w}^l \leftarrow \mathbf{w}^l - \eta \nabla L^l$

The gradient can be computed using backpropagation: $\nabla L^l = (a^l - y) f'(z^l) a^{l-1}$

Bias update: $b^l \leftarrow b^l - \eta(a^l - y)$